About the Solver Options dialog box

You can control advanced features of the solution process, load or save problem definitions, and define parameters for both linear and nonlinear problems. Each option has a default setting that is appropriate for most problems.

Max time

Limits the time taken by the solution process. While you can enter a value as high as 32,767, the default value of 100 (seconds) is adequate for most small problems.

Iterations

Limits the time taken by the solution process by limiting the number of interim calculations. While you can enter a value as high as 32,767, the default value of 100 is adequate for most small problems.

Precision

Controls the precision of solutions by using the number you enter to determine whether the value of a constraint cell meets a target or satisfies a lower or upper bound. Precision must be indicated by a fractional number between 0 (zero) and 1. Higher precision is indicated when the number you enter has more decimal places — for example, 0.0001 is higher precision than 0.01.

Tolerance

The percentage by which the target cell of a solution satisfying the integer constraints can differ from the true optimal value and still be considered acceptable. This option applies only to problems with integer constraints. A higher tolerance tends to speed up the solution process.

Convergence

When the relative change in the target cell value is less than the number in the **Convergence** box for the last five iterations, Solver stops. Convergence applies only to nonlinear problems and must be indicated by a fractional number between 0 (zero) and 1. A smaller convergence is indicated when the number you enter has more decimal places — for example, 0.0001 is less relative change than 0.01. The smaller the convergence value, the more time Solver takes to reach a solution.

Assume Linear Model

Select to speed the solution process when all relationships in the model are linear and you want to solve a linear optimization problem.

Show Iteration Results

Select to have Solver pause to show the results of each iteration.

Use Automatic Scaling

Select to use automatic scaling when inputs and outputs have large differences in magnitude — for example, when maximizing the percentage of profit based on million-dollar investments.

Assume Non-Negative

Causes Solver to assume a lower limit of 0 (zero) for all adjustable cells for which you have not set a lower limit in the **Constraint** box in the **Add Constraint** dialog box.

Estimates

Specifies the approach used to obtain initial estimates of the basic variables in each one-dimensional search.

Tangent Uses linear extrapolation from a tangent vector.

Quadratic Uses quadratic extrapolation, which can improve the results on highly nonlinear problems.

Derivatives

Specifies the differencing used to estimate partial derivatives of the objective and constraint functions.

Forward Use for most problems, in which the constraint values change relatively slowly.

Central Use for problems in which the constraints change rapidly, especially near the limits. Although this option requires more calculations, it might help when Solver returns a message that it could not improve the solution.

Search

Specifies the algorithm used at each iteration to determine the direction to search.

Newton Uses a quasi-Newton method that typically requires more memory but fewer iterations than the Conjugate gradient method.

Custom Form Properties and Methods

Custom — meaning user-defined — form properties and methods are easy to implement. If the property or method is created using the Public keyword, it is available to other modules in a project. The advantage of adding your own properties and methods to a form is that it helps to encapsulate forms; all the code the form needs is in one place, namely, part of the form module.

Adding a Custom Method

As an example, suppose you wanted to add to a form a .Center method that would center the form on the screen. You could add a Public Sub to a form that is a variation of the generalized CenterForm procedure I've used previously (see the *API Code.Bas* module in Chapter 12 "The SDK, Win32 API, and Windows Messaging System"):

```
Public Sub Center()
   Me.Move (Screen.Width - Me.Width) \ 2, _
      (Screen.Height - Me.Height) \ 2
End Sub
```

This method procedure can now be called using the dot operator (.), like any other method. To demonstrate, if the form is named Form1, and the project is set to start from Sub Main, the following code first loads and displays the form and then calls the custom method that centers the form on the screen:

```
Public Sub Main()
   Form1.Show
   Form1.Center
End Sub
```

(The sample code is saved on the companion CD-ROM as *Custom Properties and Methods.Vbp.*)

Adding a Custom Property

You can add a new property to a form simply by declaring a public variable in the form module. For example:

```
Public MyProperty As String
```